http://dx.doi.org/10.5392/IJoC.2016.12.4.053

Improved Lexicon-driven based Chord Symbol Recognition in Musical Images

Cong Minh Dinh
Department of Computer Engineering
Chonnam National University, Gwangju 500-757, South Korea

Luu Ngoc Do
Department of Computer Engineering
Chonnam National University, Gwangju 500-757, South Korea

Hyung-Jeong Yang
Department of Computer Engineering
Chonnam National University, Gwangju 500-757, South Korea

Soo-Hyung Kim
Department of Computer Engineering
Chonnam National University, Gwangju 500-757, South Korea

Guee-Sang Lee
Department of Computer Engineering
Chonnam National University, Gwangju 500-757, South Korea

ABSTRACT

Although extensively developed, optical music recognition systems have mostly focused on musical symbols (notes, rests, etc.), while disregarding the chord symbols. The process becomes difficult when the images are distorted or slurred, although this can be resolved using optical character recognition systems. Moreover, the appearance of outliers (lyrics, dynamics, etc.) increases the complexity of the chord recognition. Therefore, we propose a new approach addressing these issues. After binarization, un-distortion, and stave and lyric removal of a musical image, a rule-based method is applied to detect the potential regions of chord symbols. Next, a lexicon-driven approach is used to optimally and simultaneously separate and recognize characters. The score that is returned from the recognition process is used to detect the outliers. The effectiveness of our system is demonstrated through impressive accuracy of experimental results on two datasets having a variety of resolutions.

Key words*: Chord Symbol, Optical Music Recognition, Optical Character Recognition, Musical Image, Outlier.*

1. INTRODUCTION

Optical music recognition (OMR) [1] has been a field of interest for several centuries. The motivation is to essentially be able to store ancient musical scores and to build musical databases with numerous entries for use in musicology [2]. A number of systems have been extensively developed. However, most of these have focused on musical symbols, such as notes, rests, sharps, etc., and have disregarded chord symbols that provide information of the harmony in the music sheet. A chord consists of multiple voices that are sustained or move in parallel. A chord symbol is composed of one or a couple of the following parts in order: the root note (e.g. A, B or C); the chord quality (e.g. major, maj, or M); the number of an interval (e.g. seventh, or 7); the altered fifth (e.g. sharp five or #5); an additional interval number (e.g. add 13 or add13) [11]. Chords are extensively used in all styles of tonal music, serving many variations of jazz, pop, rock, theater music, etc. These could be indicated by placing notes on a stave or through the use of symbols (chord symbols). Since these chord symbols can be found in printed sheet music, jazz fake books and production

* *Corresponding author, Email: hjyang@jnu.ac.kr*
Manuscript received Sep. 06, 2016; revised Dec. 21, 2016; accepted Dec. 22, 2016

scores used in the recording industry, recognizing these cannot be avoided.

Even though optical character recognition (OCR) systems [3], [4] can recognize the characters after excluding all other musical symbols on staves, it is not simple when other symbols appear above the staves and when lyrics are on the same music sheet along with the chord symbols. Some lyrics are presented as a mixture of different languages, such as English and Korean. This causes multiple-language problem [3] that increases the complexity level of the OCR system. The inclusion of other symbols, such as dynamics, ornaments, codas, etc., also causes much confusion. Therefore, we need a system that more effectively recognizes chord symbols and considers the others as outliers. In addition, there is one more problem when capturing images in that the captured images are usually distorted or slurred. The characters become deformed, and those that touch together after binarization are especially difficult cases to be recognized. Therefore, it is considered un-distortion and separation for the cases where symbols touch each other.

Kodirov, et al. [5] proposed a good approach to overcome the touching cases by using re-binarization and rule-based separation. Separation is improved in this approach, but the characters are sometimes broken. The result of the separation is also very sensitive in terms of the choice of the area of interest because it is hard to take all cases into account, especially with the appearance of special characters including '(', ')' or '/'. Moreover, outliers that cannot be avoided in practice were not considered. In addition, it has not tried long length of chord symbols which are over three characters and include special symbols, such as '(', ')', '-' or '/'. In [6], Kim presented over-segmentation and lexicon-driven approach which vertically slices an image into small segments with a width determined by the constant value. The optimal segment combination is then found from the slices of the image by using a lexicon-driven word scoring technique and a nearest-neighbor classifier. This combination provides the final segmentation positions for the individual characters in the image and the best matching word in the lexicon simultaneously. However, the parameters in the method are fixed and sensitive with changes of the resolution. Long chord symbols (of over five characters) cannot be processed using this approach because the number of combinations is too large to be handled by this method. Moreover, many combinations are invalid, unnecessary and cause the segment optimization to be incorrect. In addition, it cannot completely avoid the appearance of the outliers, such as lyrics and other symbols.

In this paper, we propose an approach which is more appropriate and effective in addressing these issues. After binarization, un-distortion, stave detection, stave information extraction, stave-line removal, and lyrics removal, candidates for the chord symbols are extracted by using a rule-based method. The connected components are grouped together to build candidates, and finally, these candidates are passed to a lexicon driven recognition system. It uses over-segmentation and finds the optimal segment combination by implementing a lexicon-driven word scoring technique and a nearest-neighbor classification. The proposed method is adaptively adjusted to the resolutions. The recognition is performed on each connected component instead of over the whole chord symbol to reduce the number of possible combinations. Invalid combinations are filtered out in order to reduce the processing time and increase accuracy. A new scoring method that more effectively describes dissimilarity in the outliers to the templates is proposed for outlier detection.

This paper is organized as following. Section 2 presents the details of our system. Subsection 2.1 provides the preprocessing stage. Subsection 2.2 explains the improvements that are applied to the recognition system. The experiment results that prove the effectiveness of the proposed method are described in Section 3. Finally, Section 4 contains the conclusions and some discussions.

2. PROPOSED SYSTEM

The proposed system includes the following steps. First, a captured image is preprocessed. Then, the candidates of the chord symbols are detected, and finally, an improved lexicon driven strategy is used to recognize the chords and to detect outliers. In this paper we focus on the chord recognition so that the preprocessing step is explained shortly referring to other literatures.

2.1 Chord symbol region detection

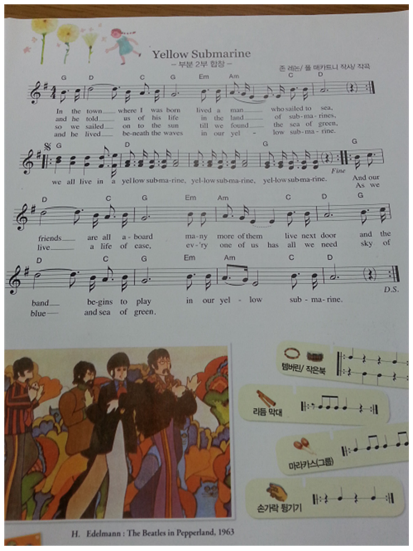
The preprocessing stage includes binarization, un-distortion, stave removal, lyric removal and chord symbol candidate detection. After binarization, the image is undistorted similarly as [7]. The stave lines and the bar lines are detected. Each stave area, then, is divided into local regions according to these bar lines. Then, the image is corrected for each of these regions using the information of the bar lines. Stave detecting and removing is conducted as [8]. Stave-line height, stave space, stave line positions and stave pixels are obtained. Then, all stave pixels are straightforwardly removed.

The lyrics are excluded to reduce the outliers for the chord symbol recognition by employing the methods proposed in [9], [10]. The baselines are detected first according to the local minima of the connected components. The regions of the lyric lines are reconstructed after the heights of the lines have been estimated by the components that are connected to the baselines. It handles the touching and overlapping cases by cropping out the parts of the connected components that are out of the line regions. After the image has been corrected and cleaned up, only chord symbols and some outliers remain above the stave area.

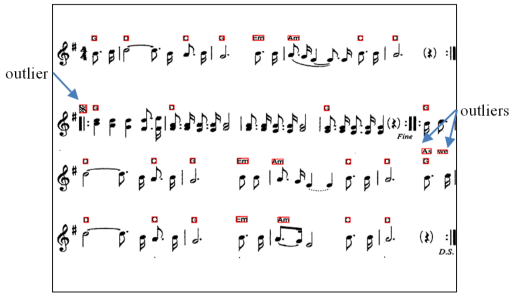
A chord symbol always starts by a captained letter. In addition, the chord symbol is similar to a word in natural language in that there are small spaces between the parts. It is possible to detect the candidate of a chord symbol by considering the width and the height of the components and the spaces between them. Therefore, the touched parts which are the result of a distortion or a slur of a captured image after binarization should be considered.

Assume that we already have the stave space d_s and the height h_s of the stave lines from the stave detection stage. The set of N components c_i, $C = \{c_i \mid l_i \le l_j, 1 \le i < j \le N\}$ above

the stave area is sorted from left to right according to the left bound l_i. Each component c_i has left bound l_i, top bound t_i, bottom bound b_i, width w_i and height h_i. The process for this stage is described in the algorithm 1. A is the set that stores chord symbol candidates.



(a)



(b)

Fig. 1. (a) Original image (b) Result of preprocessing stage

N_A is the number of the candidates in A. B_{N_A+1} is the boundary of the (N_A+1)-th chord symbol candidate. $B^r{}_{N_A+1}$, $B^t{}_{N_A+1}$ and $B^b{}_{N_A+1}$ are the right, top and bottom of B_{N_A+1} respectively. The parameters α_1, α_2, β_1 and β_2 play the role to check for the size of the first component of a candidate (with α_1 and α_2 for the width, β_1 and β_2 for the height). From the original image Fig. 1(a), we have the result of this stage illustrated in Fig. 1(b), and the candidates are marked with red rectangles. As we can see, there are some outliers that we have to detect in the next stage.

Algorithm 1: Detection of the Chord Symbol Candidates

Data: $C = \{c_i \mid l_i \le l_j, 1 \le i < j \le N\}$, d_s and h_s

Result: Set of chord symbol candidates A

begin

$A \leftarrow \square$; $N_A \leftarrow 0$; $i \leftarrow 1$;

while $i \le N$ **do**

if c_i *has not been stored,* **and** $w_i \in (\alpha_1 \times d_s, \alpha_2 \times d_s)$ **and** $h_i \in (\beta_1 \times d_s, \beta_2 \times d_s)$ **then**

Include c_i into the (N_A+1)-th candidate;

$B_{N_A+1} \leftarrow$ the boundary of c_i;

$j = i + 1$;

while $j \le N$ **do**

if c_j *has not been stored,*

and $(w_j < \alpha_2 \times d_s) \cap (l_j - B^r{}_{N_A+1} < d_s)$

and $MIN(b_j, B^b{}_{N_A+1}) - MAX(t_j, B^t{}_{N_A+1}) > 0.5 \times h_s$

and $((w_j > h_s) \cup (h_j > d_s))$ **then**

Include c_j into the $(N_A + 1)$-th candidate;

Update B_{N_A+1} to cover c_j;

end-if

$j = j + 1$;

end-while

end-if

$i = i + 1$;

end-while

end

2.2 Chord Symbol Recognition

The candidates of the chord symbols are detected in the above stage are passed to the recognition system. This method is very effective for cases where symbols are touching. The goal of such system is to determine whether each of the candidates is a chord symbol or an outlier, and what the parts are (root note, chord quality, etc.) if it is a chord. First, we extract the sub-image of the candidate by copying the components to a new image. One sub-image for each candidate is extracted from the entire image of the music sheet. These sub-images have been input for the recognition system, and a list of chord names is pre-defined and plays the role of a dictionary (lexicon), according to the description of the chord symbol structure as in [11]. The output offers the best separation, the best matching word (chord symbol) and score. The score describes the distance or the dissimilarity of the candidate to the templates and is used for outlier detection during the final step.

In this paper, we tackle some issues from the over segmentation and lexicon driven approach as follows:

• First it should be considered the sensitivity to the font and resolution. The changes in the resolution or the font, as in Fig. 2(a), make changes in the stroke width of the characters. Thus, the segment points are hard to be at the touching points for these cases.

• Second, in the process of finding the optimal combination, some groups, as shown in Fig. 2(b), that are created by the segments in the middle of the character or two of the characters produce incorrect results. These are therefore referred to as invalid groups.

• Third, some special characters including a slash (/), minus (-), or left [(] and right [)] brackets have a size that is insufficient for the feature vector extraction (for example, with dimensions of 4 by 9 by 7) due to the inadequate resolution. The group image is nonlinearly divided into a mesh (for example, 9 by 7) in four directions. However, the size of the group image is sometimes too small (for example, less than 9 by 7), as shown in Fig. 2(c).

• Forth, the long chord symbols (those with as many characters as 4 or 5, as shown in Fig. 2(d)) generate numerous combinations. This causes the computation time increases exponentially.

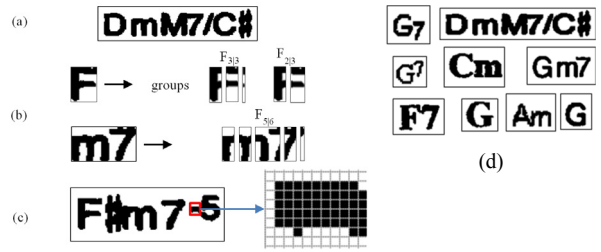
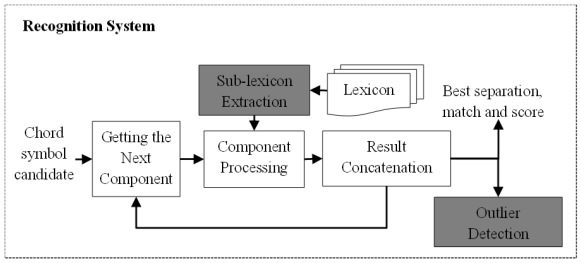


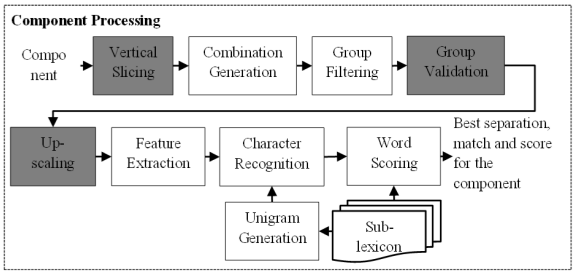
Fig. 2. Examples for issues: (a)Variety of font and size; (b) Some invalid groups (the second and the third in *F*3|3, and the second in *F*2|3 of 'F'; the third in *F*5|6 of 'm7') have more than one component; (c) Small character '-'. Its height is only 6 pixels, less than 9; (d) long chord symbols (7 characters)

Fifth, outliers always appear in practice because the lyric removal is not perfect for all cases. Moreover, the other symbols, such as dynamics, ornaments, etc., often appear on the top of staves with a size that is similar to a chord symbol. Thus, the chord symbol candidate detection cannot ignore these, as shown in Fig. 1(b).

Therefore, we propose a new approach for recognition as in Fig. 3(a) to resolve the above issues. In order to solve the first issue, we modified the vertical slicing method of [6]. The second and third issues are handled by group validation step and up-scaling step as shown in Fig. 3(b) respectively. For the forth issue we propose a sub-lexicon extraction step. Finally, we proposed an outlier detection method to solve the fifth issue.



(a)



(b)

Fig. 3. (a) Recognition system structure. (b) Structure of component processing step

After the image of the candidate has been input, the connected components are extracted using 8-connectivity. We perform recognition for each component instead of the entire image through the use sub-lexicon extraction. In particular, these components are iteratively treated in the Component Processing step (as illustrated in Fig. 3(b)). In this step, the full lexicon is used only to recognize the first component. All of the following ones use the sub-lexicon extracted in full (the details are given in the Subsection 2.2.4). The output of this step is the best separation, the best match and the score for each component. The best matches for all components are concatenated, and the scores of these are passed to the outlier detection step in order to determine whether the candidate has a chord symbol or an outlier. If it is a chord symbol, the content is then the concatenation of the best matches.

2.2.1 Vertical Slicing

In this step, we propose an adaptive slicing method based on histogram and the adaptive value of segment point. In Ref. [6], the value of the segment point is hard-set to 10 pixels and the stroke width of the characters can be of three to four pixels, depending on the resolution or font. This causes the second issue in that the slicing points can be far from the touching points. Therefore, the value of segment point should not be fixed. In particular, it is adjusted according to the stroke width of the characters. However, this should not be too small due to the number of segments that then become large. The value of segment point *SEG_T* is thus determined according to the following function:

$$SEG\_T = \begin{cases} \lfloor st\_w \rfloor, st\_w > 3 \\ 3, o.w \end{cases} \quad (1)$$

where *st_w* is the average value of the stroke width that is computed by *st_w* = *S*/*a*. *S* is the total number of black-pixels in the component, *a* is the number of pixels that their right neighbors, bottom neighbors and bottom-right neighbors are all black.

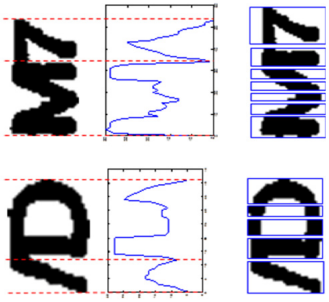


Fig. 4. Illustration of vertical slicing method using vertical projection histogram. First column is original images of chord symbols "/D" and "M7" with touches. Second column is vertical projection histograms. The last column is the results of the vertical slicing.

Even though the segment point *SEG_T* is adapted to the average stroke width, it sometimes segments the touching

points in the wrong way because of the variation in the font and size, as shown in Fig. 2(a). Therefore, we use another method for slicing. As observed, the touching points are often the minima of the vertical projection histogram, as shown in Fig. 4 (first and second columns). Thus, we perform the slicing according to this histogram as follows. The local minima are determined from the left to right of the histogram, and two continuous minima that have a distance larger than or equal to the segment point *SEG_T* (defined as in Equation 1) create a segment. The result of the new slicing method is shown in Fig. 4 (the last column).

2.2.2 Group Validation

In the second issue, an invalid group is one that is not a character. In detail, there are two cases of this as Fig. 2(b). The first one involves a group of segments in the middle of the character. The second one involves a group of segments of two characters. They have some specific characteristics as the following. For the first case, the group has more than one connected components with a small height. Otherwise, the group in the second case has more than two connected components or has only two with the additional height. Assume that we have a group with a set of components $C = \{c_i\}$, where c_i is the *i*-th component in the group with height h_i and h is the height of the group. We summarize these cases by following rules:

- Rule GV1: $|C_1| > 1$, where $C_1 = \{c_i \mid h_i < 0.5 \times h\}$
- Rule GV2: $|C_2| > 1$, where $C_2 = \{c_i \mid h_i > 0.7 \times h\}$

where the operator |.| denotes the number of elements in a set. When a group satisfies one of the above rules, it is considered to be invalid and is ignored from the next sub-steps.

2.2.3 Up-scaling small groups

The feature extraction method in [6] is similar to the zoning method. The group image is divided into a mesh with fixed sized of 9 by 7 in four directions. However, in the case with a low resolution, the size of the groups that correspond to some special characters, such as a slash, minus, left or right brackets, is small (the forth issue). Their width is smaller than *seven* or the height is smaller than *nine*. This makes the feature vector incorrect. Therefore, before the extraction process, we have to up-scale these so that their width and height are respectively equal or larger than 9 and 7. Assume that the size of the group is h_g by w_g. The scaling coefficient k is calculated as follows:

$$k = \begin{cases} MAX(\frac{9}{h_g}, \frac{7}{w_g}), if\,(h_g < 9) \cup (w_g < 7) \\ 1, o.w \end{cases} \quad (2)$$

2.2.4 Sub-lexicon extraction

In order to solve the forth issue, we consider each component once and the lexicon that is used for each is a part of the whole. It is extracted from the full lexicon according to the results of the previous components. Only the words that begin with the concatenation of the matches of the previous components are extracted. However, these are excluded from the beginning parts that have been already matched. For example, a candidate has two components. The first one is already matched to the character `B' and the full lexicon consists of the words: `Ab', `B7' and `B#'. Then, the sub-lexicon that is used to recognize the second one only contains words that begin with `B', but `B' is excluded. In particular, the sub-lexicon is {`7', `#'}.

2.2.5 Outlier Detection

For the last (fifth) issue, we notice that after the step Component Processing, we have scores that show the distance or the dissimilarity between the components and the templates. The components of the outlier often have large scores over a threshold τ. Thus, the average or the largest one of these scores are chosen in order to make the final score for the whole candidate and then use it to detect outliers. However, outliers sometime have one or two components with a small score due to the noise or as a result of the segmentation optimization. This produces an average that is sometimes small and the outlier becomes such as a chord symbol. Moreover, the chord symbol sometimes has one component with a high score due to noise or special characters, such as a flat, sharp, slash, minus, in the case where these touch other characters too much and are deformed after separation. Such cases often include an unnecessary part from the other characters or a lack of a part because of their special size (which is smaller than others) and position (higher or lower than usual). Then, the final score is large if we choose the biggest one and the chord symbol is excluded as an outlier.

Therefore, we choose another way to make the final score to avoid these problems as follows. The second maximum in the scores of the components is chosen as final if the number of components is larger than one. Otherwise, only one score is chosen. Assume that the candidate has *n* components with their corresponding scores $(d_1, d_2, ..., d_n)$. The final score p is described by following equation:

$$p = \begin{cases} SECOND\_MAX(d_1, d_2, ..., d_n), n > 1 \\ d_1, n = 1 \end{cases} \quad (3)$$

where *SECOND_MAX(.)* is the function that extracts the second maximum of a list of numbers.

3. EXPERIMENTS

The performance of the proposed system is evaluated on two datasets of Korean printed music. Dataset A contains over two hundred captured images including those distorted with a high resolution (an average size of 2448 by 3264). Dataset B contains over one hundred scanned-printed images with a low resolution (an average size of 1225 by 1396) and without distortion. The ground truth dataset is manually constructed and the correct results of the chord symbol candidate detection are considered, with these containing only either parts of chord

symbols or outliers. These datasets include all issues that have been previously mentioned, such as the variation in stroke width, the cases where characters are touching, long chord symbols (with a length larger than 3) and chord symbols with special characters (‘/’, ‘-’, ‘(’ and ‘)’), as shown in Table 1. There are thirty character classes, including: `1', `2', `3', `4', `5', `6', `7', `9', `a', `A', `B', `C', `d', `D', `E', `F', `g', `G', `i', `j', `m', `M', `s', `u', flat, sharp, ‘(’, ‘)’, ‘-’ and ‘/’. Each class has twenty-five templates collected from Web with each not being related to those in the datasets. The details of the datasets are given in Table 1.

Table 1. Details of Dataset

Dataset	No. chord symbols	No. long chords	No. characters	No. special characters	No. touching characters	Min. Stroke width (pixel)	Max. Stroke width (pixel)	No. outliers
A	4935	273	8925	356	444	3.29	9.17	2161
B	1327	3	1660	2	203	1.78	4.92	153

The proposed method is compared to that described by Kim, et al. [6] and one-class classifiers, such as Gaussian Data Description (GAUSSDD) [12], k-Means Data Description (KMEANSDD) [12], Minimum Spanning Tree Data Description (MSTDD) [13], k-Nearest Neighbor Data Description (KNNDD) [14], Support Vector Data Description (SVDD) [15], Linear Programming Data Description (LPDD) [16] and Minimax Probability Machine (MPMDD) [17]. For the method [6], we use the word score that is the result of finding the best match for the outlier detection, and *SEG_T* is assigned to integer numbers in the range [5], [15]. With respect to the one-class classifiers, we use the dissimilarity degree to make the score, and the final score for the candidate is the average of the scores of the components (connected components). Their parameters are determined by the grid-search method, and a cross-validation method is applied in their training process.

In order to compare the performance of the outlier detection between the proposed method and the other methods, we use Receiver Operating Characteristic (ROC) curves [18] and the area under the ROC-curve (AUC) [19]. This integrates the fraction of the true positive over varying thresholds. Higher values indicate a better separation between the targets (chord symbols) and outliers. To show how good chord symbols are recognized, we use precision, recall and an equation that can be referred to as the character accuracy [20] that counts insertion, deletion and substitution errors. For counting these errors, the alignment algorithm proposed by Needleman and Wunsch [21] is applied. To see how much the confusion between classes, we perform the measure proposed in [22] that shows the accuracy of multi-class classification by an equation on the elements of the confusion matrix.

As shown in Fig. 5, Table 2 and Table 3, the proposed method achieves an impressive performance for most of kinds of evaluations and for both datasets A and B while traditional methods (Table 2, Table 3 and Fig. 5(a,g)) and the method provided by Kim, et al. [6] (Fig. 5(b-f,h-l)) have a lower accuracy, especially for dataset B. The method [6] offers good performance in terms of the outlier detection (Fig. 5(b,h)), but its recognition accuracy is worse and there are more mistakes. In particular, its results are sensitive to the value of the constant *SEG_T* and the variety of resolutions. For dataset A, it works well with *SEG_T = 10*, but it is good with *SEG_T = 5* for dataset B. Moreover, its performance deteriorates on dataset B when we just change *SEG_T* a little. For example, its multi-class classification accuracy decreases by 13.67% (from 0.9125 to 0.7878) when we change *SEG_T* from 5 to 7, that is, a difference of just two pixels. On the other hand, the accuracy of the proposed method is still high, even when dataset B has a low resolution. In the Fig. 5(h), the AUC of the proposed method is lower than the method [6] at a few points but the difference between 0.9925 (the proposed method) and 0.994 ([6]) is insignificant. Importantly, we do not have to adjust any constant.

Table 2. Results for dataset A comparing the proposed method with one-class classifiers in AUC, precision, recall, character accuracy and multiclass classification accuracy

Method	AUC	Precision (%)	Recall (%)	Character Accuracy (%)	Multiclass Classification Accuracy
GAUSSDD	0.9033	84.75	80.41	74.89	0.8147
KMEANSDD	0.9498	84.57	81.37	75.39	0.8186
MSTDD	0.9654	93.84	86.30	83.60	0.8900
KNNDD	0.9579	88.32	83.03	78.64	0.8420
SVDD	0.9327	85.36	82.68	74.60	0.8387
LPDD	0.9128	82.38	79.62	71.14	0.8125
MPMDD	0.9450	86.84	80.72	75.71	0.8224
Proposed	**0.9988**	**99.56**	**99.54**	**99.15**	**0.9984**

Table 2. Results for dataset B comparing the proposed method with one-class classifiers in AUC, precision, recall, character accuracy and multiclass classification accuracy

Method	AUC	Precision (%)	Recall (%)	Character Accuracy (%)	Multiclass Classification Accuracy
GAUSSDD	0.9033	73.69	67.83	63.37	0.5993
KMEANSDD	0.9093	73.61	67.71	63.31	0.5968
MSTDD	0.8922	81.30	74.64	70.24	0.6161
KNNDD	0.9271	80.46	73.92	69.52	0.6608
SVDD	0.7995	60.56	55.78	51.27	0.3572
LPDD	0.8443	69.79	64.16	59.76	0.5341
MPMDD	0.8979	72.15	66.33	61.93	0.4678
Proposed	**0.9922**	**97.49**	**93.73**	**93.49**	**0.9542**

4. CONCLUSION

In this paper we have proposed an improved lexicon driven approach to detect chord symbol candidates by using information from the staves and performing matching through the use of a recognition system. The main idea is that we have simultaneously performed segmentation and matching to overcome cases where symbols are touching, and we use the

matching scores to exclude outliers. After the image has been undistorted, the staves and lyrics are removed and the candidates for the chord symbols are detected. These are recognized by the over segment based recognition system. The results we have obtained prove that our system is effective and provides a very high performance (more than 0.99 of AUC with two datasets), presenting it works for difficult situations, such as, when the image is distorted, the resolution varies, the characters touch together and many outliers appear.

For the future research, we will investigate more when the components are broken after binarization. That is, one component becomes many. Then, the chord symbol candidate detection will provide wrong results (such as the component lacks some parts) or the recognition system that processes component by component will provide the wrong match.

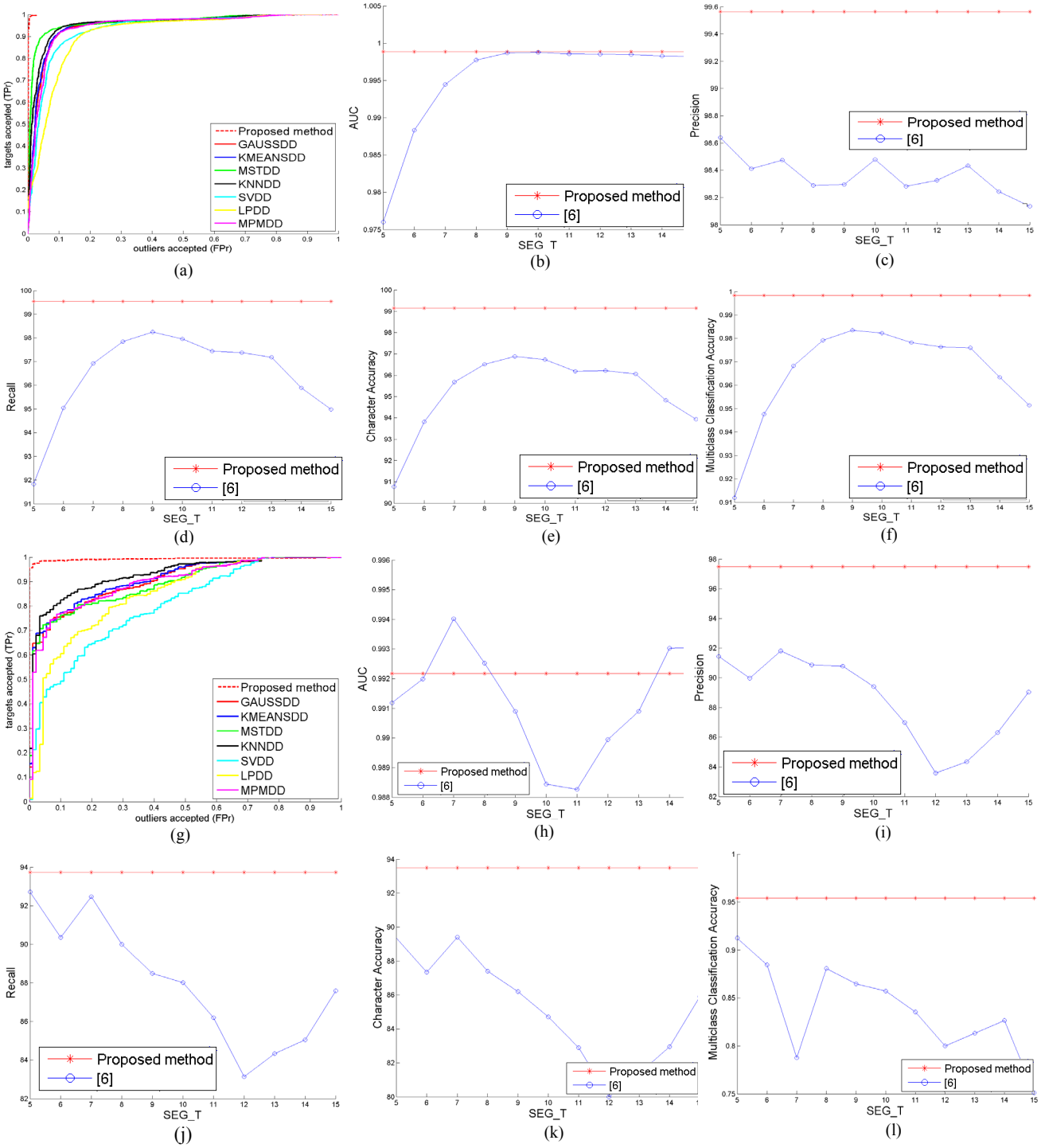


Fig. 5. Results for dataset A (a-f) and B (g-l). (a, g) ROC curves compare the proposed method with one-class classifiers. (b, h), (c, i), (d, j), (e, k) and (f, l) that are respectively AUC, precision, recall, character accuracy and multi-class classification accuracy graphs compares the proposed method with the method proposed by Kim, et al. [6] at every integer number of SEG_T in the range [5], [15]. The graph of the proposed method is horizontal because it is independent to the constant SEG_T.

REFERENCES

[1] L. Pugin, "Optical Music Recognition of Early Typographic Prints using Hidden Markov Models," Proceedings of the 7th International Conference on Music Information Retrieval, Oct. 2006, pp.53-56.

[2] J. C. Pinto, P. Vieira, M. Ramalho, M. Mengucci, P. Pina, and F. Muge, "Ancient Music Recovery for Digital Libraries," in Research and Advanced Technology for Digital Libraries," J. Borbinha and T. Baker, eds. Springer Berlin Heidelberg, vol. 1923, 2000, pp. 24-34.

[3] E. Borovikov, *A survey of modern optical character recognition techniques*, 2014.

[4] V. K. Govindan and A. P. Shivaprasad, "Character recognition - A review," Pattern Recognition., vol. 23, no. 7, 1990, pp. 671-683.

[5] E. Kodirov, S. Han, G. S. Lee, and Y. C. Kim, "Music with Harmony: Chord Separation and Recognition in Printed Music Score Images," Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication, 2014, pp. 1-8.

[6] S. H. Kim, S. Jeong, and C. Y. Suen, "A lexicon-driven approach for optimal segment combination in off-line recognition of unconstrained handwritten Korean words," Pattern Recognit., vol. 34, no. 7, 2001, pp. 1437-1447.

[7] Q. N. Vo, T. Nguyen, S. H. Kim, H. J. Yang, and G. S. Lee, "Distorted Music Score Recognition without Staff line Removal," Pattern Recognition (ICPR), 2014 22nd International Conference on, Aug. 2014, pp. 2956-2960.

[8] B. Su, S. Lu, U. Pal, and C. L. Tan, "An Effective Staff Detection and Removal Technique for Musical Documents," Document Analysis Systems (DAS), 2012 10th IAPR International Workshop on, Mar. 2012, pp. 160-164.

[9] J. A. Burgoyne, Y. Ouyang, T. Himmelman, J. Devaney, L. Pugin, and I. Fujinaga, "Lyric extraction and recognition on digital images of early music sources," Proceedings of the 10th International Society for Music Information Retrieval Conference, vol.10, 2009, pp. 723-727.

[10] M. Feldbach and K. D. Tonnies, "Line detection and segmentation in historical church registers," Document Analysis and Recognition, 2001 Proceedings, Sixth International Conference on, 2001, pp. 743-747.

[11] R. Gorow, *Hearing and Writing Music: Professional Training for Today's Musician, 2nd ed.*, September Publishing, 2000.

[12] Q. H. Wang, L. S. Lopes, and D. M. J. Tax, "Visual Object Recognition Through One-Class Learning," in Image Analysis and Recognition, Springer Berlin Heidelberg, vol. 3211, 2004, pp. 463-470.

[13] P. Juszczak, D. M. J. Tax, E. Pękalska, and R. P. W. Duin, "Minimum spanning tree based one-class classifier," Neurocomputing, vol. 72, no. 7-9, 2009, pp. 1859-1869.

[14] D. M. J. Tax, *One-class Classification*, 2001.

[15] D. M. J. Tax and R. P. W. Duin, "Support vector domain description," Pattern Recognition. Letter, vol. 20, 1999, p. 1191.

[16] E. Pekalska, D. M. J. Tax, and R. P. W. Duin, *One-Class LP Classifiers for Dissimilarity Representations*, in Advances in Neural Information Processing Systems 15, MIT Press, 2003, pp. 777-784.

[17] G. R. G. Lanckriet, L. E. Ghaoui, and M. I. Jordan, *Robust Novelty Detection with Single-Class MPM*, in Advances in Neural Information Processing Systems 15, MIT Press, 2003, pp. 929-936.

[18] T. Fawcett, "An Introduction to ROC Analysis," Pattern Recognition. Letter, vol. 27, 2006, pp. 861-874.

[19] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," Pattern Recognit., vol. 30, no. 7, 1997, pp. 1145-1159.

[20] T. Nartker, K. Taghva, R. Young, J. Borsack, and A. Condit, "OCR correction based on document level knowledge," International Symposium on Electronic Imaging Science and Technology, vol. 5010, 2003, pp. 103-110.

[21] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," J. Mol. Biol., vol. 48, no. 3, Mar. 1970, pp. 443-453.

[22] A. B. David, "Comparison of classification accuracy using Cohen's weighted kappa," Expert Syst. Appl., vol. 34, 2008, pp. 825-832.

Cong Minh Dinh

He received his B.S. degree in Mathematics & Computer Science from Ho Chi Minh University of Science, Vietnam in 2008, was a software developer in eSilicon Corporation from 2009 until 2012, and received his M. Eng. Degree in Electronics & Computer Engineering at Chonnam National University, Korea in 2015. His research interests are Data Mining and Pattern Recognition.

Luu Ngoc Do

He received the B.S and M.S from Chonnam National University, South Korea in 2013. He is currently a Ph. D student at Dept. of Electronics and Computer Engineering, Chonnam National University, South Korea. His main research interests include Data Mining, Pattern Recognition, Machine Learning and Image Processing.

Hyung-Jeong Yang

She received her B.S., M.S. and Ph.D. degrees from Chonbuk National University, Korea. She was a Post-doc researcher at Carnegie Mellon University, USA. She is currently a professor at Dept. of Electronics and Computer Engineering, Chonnam National University, Gwangju, Korea. Her main research interests include multimedia data mining, pattern recognition, artificial intelligence, e-Learning, and e-Design.

Soo-Hyung Kim
He received his B.S. degree in Computer Engineering from Seoul National University in 1986, and his M.S. and Ph.D. degrees in Computer Science from Korea Advanced Institute of Science and Technology in 1988 and 1993, respectively. From 1990 to 1996, he was a senior member of research staff in Multimedia Research Center of Samsung Electronics Co., Korea. Since 1997, he has been a professor in the Department of Computer Science, Chonnam National University, Korea. His research interests are pattern recognition, document image processing, medical image processing, and ubiquitous computing.

Guee-Sang Lee
He received his B.S. degree in Electrical Engineering and his M.S. degree in Computer Engineering from Seoul National University, Korea in 1980 and 1982, respectively. He received his Ph.D. degree in Computer Science from Pennsylvania State University in 1991. He is currently a professor of the Department of Electronics and Computer Engineering in Chonnam National University, Korea. His research interests are mainly in the field of image processing, computer vision and video technology.